

3. With regard to the independent claims 1, 11, 21, and 31, the Examiner asserts that Tucker teaches a method for providing mutual exclusion (col. 1, line 58) for a resource by a first process [52, fig. 3] in a computer system having a plurality of processes involving receiving an inquiry from a second process inquiring whether said first process (52, try to acquire mutex lock) owns said resource [col. 4, lines 7-12]. The Examiner also asserts that, while Tucker does not explicitly teach determining an owner process for said resource other than said first process and creating a lock for said resource indicating said owner process, Tucker teaches "if the lock is not acquired (e.g. the lock is held by other prior scheduled thread), the thread owning the particular mutex lock is identify 54" [col. 4, lines 12-14]. The Examiner further asserts that the second process can "acquire the mutex lock by repeating the step 52-56" [col. 4, lines 17-18], and the second thread can attempt to acquire the lock again through the same process and lock the resource when the lock is available. The Examiner concludes that "it would have been obvious to one ordinary skill in the art at the time of the invention was made to acquire lock, the second process must identify the owner of the lock for requesting to access the resource."

4. Applicant respectfully disagrees with the Examiner's assertions and conclusion regarding Tucker and its relevance to the present invention of claims 1, 11, 21, and 31 on a number of grounds.

In the method of Tucker, a mutex lock exists for a particular resource. With reference to FIG. 3, the second process attempts to acquire the mutex lock and thereby acquire the corresponding resource [52]. If the second process successfully acquires the mutex lock [in 53], then the second process stores information about itself in the mutex lock [53a] (this information is used by subsequent processes that attempt to acquire the mutex lock). If the second process is unable to acquire the mutex lock [in 53], then the second process gets the current lock owner from the mutex lock [54] and looks up the current state of the owner [55]. If the owner is currently running [in 56], then the second process

continues trying to acquire the mutex lock by looping back to [52] (i.e., "spinning"). If the owner is not currently running [in 56], then the second process is put to sleep (i.e., is "blocked") [56a] until such time that the lock is available and the second process is awakened. It should be noted here that the mutex lock must be in existence before any process tries to acquire the mutex lock using the method of FIG. 3, which implies that the mutex lock is established in advance. It should also be noted that the mutex lock is accessed by all processes trying to acquire the resource.

In the present invention of claims 1, 11, 21, and 31, the first process receives an inquiry from the second process inquiring whether the first process owns a resource. Since the first process "knows" that it does not own the resource, the first process determines the owner of the resource (which, by implication, will be the second process in a two process system, although it could be a third process in a system having more than two processes) and creates a lock for the resource indicating the owner process for the resource. It should be noted here that the lock is not created by the first process until after it receives the inquiry from the second process. It should also be noted that, in accordance with the present invention, the lock created by the first process is only used by the first process (not by the second process or a third process). In the distributed environment of the present invention, the creation of the lock by the first process allows the first process to determine the status and owner of the resource locally without having to send inquiries to other processes (which are time-consuming in a distributed environment).

As discussed above, the Examiner asserts that Tucker teaches a method for providing mutual exclusion (col. 1, line 58) for a resource by a first process [52, fig. 3] in a computer system having a plurality of processes involving receiving an inquiry from a second process inquiring whether said first process (52, try to acquire mutex lock) owns said resource [col. 4, lines 7-12]. Clearly, in Tucker, the second process (i.e., the acquiring process) never sends an inquiry to the first process (and therefore the first process never receives an inquiry from

the second process) inquiring whether the first process owns the resource. This is true even if the first process is the current owner of the resource. Thus, the Examiner's assertion that Tucker teaches a method for providing mutual exclusion (col. 1, line 58) for a resource by a first process [52, fig. 3] in a computer system having a plurality of processes involving receiving an inquiry from a second process inquiring whether said first process (52, try to acquire mutex lock) owns said resource [col. 4, lines 7-12] appears to be incorrect.

Also as discussed above, the Examiner concludes that " it would have been obvious to one ordinary skill in the art at the time of the invention was made to acquire lock, the second process must identify the owner of the lock for requesting to access the resource." In Tucker, it is the second process (i.e., the process trying to acquire the mutex lock) that identifies the lock owner [54] when the mutex lock is owned by another process, not for the purpose of requesting access to the resource, but rather for the purpose of determining whether or not the owner process is running [55 and 56] (the second process does not acquire the resource in either case since it is owned by another process). In the present invention of claims 1, 11, 21, and 31, however, it is the first process (i.e., the non-acquiring process) that identifies the owner of the lock, not for the purpose of requesting access to the resource, but rather for the purpose of storing the owner information in the lock created by the first process so that the first process can subsequently determine the status and owner of the resource locally without having to send inquiries to other processes. Thus, the Examiner's conclusion that " it would have been obvious to one ordinary skill in the art at the time of the invention was made to acquire lock, the second process must identify the owner of the lock for requesting to access the resource" does not appear to be relevant to a determination of patentability because (1) the second process of Tucker does not identify the owner of the lock for requesting access to the resource; (2) it is the first process and not the second process of the present invention that identifies the owner of the resource; and (3) the first process of the present

invention does not identify the owner of the resource for requesting access to the resource.

Also as discussed above, Tucker requires that the lock be in existence before the second process can try to acquire the resource, but in the present invention of claims 1, 11, 21, and 31, the lock is not created until after the second process tries to acquire the resource by sending an inquiry to the first process. Thus, Tucker appears to teach away from the present invention in this respect.

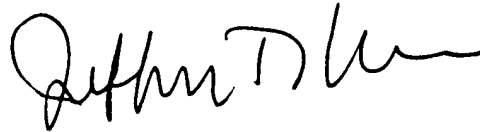
Applicant therefore respectfully submits the present invention of claims 1, 11, 21, and 31 are not obvious in view of Tucker and are patentable over Tucker. Tucker does not teach or otherwise suggest a method for providing mutual exclusion for a resource involving a first process receiving an inquiry from a second process inquiring whether said first process owns said resource, determining an owner process for said resource other than said first process, and creating a lock for said resource indicating said owner process. It would not have been obvious to one skilled in the art at the time of the invention for the second process to send an inquiry to the first process regarding ownership of a resource and for the first process to determine an owner process for the resource and create a lock for the resource indicating the owner process, based upon Tucker.

5. Applicant respectfully submits that, in light of the above discussion, the Examiner's rejections of all dependent claims are moot. Applicant believes that independent claims 1, 11, 21, and 31 are in a form suitable for allowance. Because a dependent claim is deemed to include all limitations of its parent claim and any intervening claims, all dependent claims are also believed to be in a form suitable for allowance.

6. Claims 1-40 are pending in this application. All pending claims are believed to be in a form suitable for allowance. Therefore, the application is believed to be in a condition for allowance. The Applicant respectfully requests

early allowance of the application. The Applicant requests that the Examiner contact the undersigned, Jeffrey T. Klayman, if it will assist further examination of this application.

Respectfully submitted,



Jeffrey. T. Klayman
Attorney for Applicant
Registration No. 39,250

BROMBERG & SUNSTEIN LLP
125 Summer Street
Boston, MA 02110-1618
(617) 443-9292

204509

